IN THE CLAIMS

Please amend claims 1, 3, 4, 9, 11, 12, 17, 19 and 20 as indicated below.

1. (Currently Amended) A <u>branch prediction</u> method comprising:

    <u>receiving a fetch address;</u>

    detecting a first level cache does not contain a first branch prediction information corresponding to ~~a first~~ <u>the fetch</u> address;

    determining whether a second level cache contains a second branch prediction information corresponding to said ~~first~~ <u>fetch</u> address, said second branch prediction information comprising a subset of said first branch prediction information;

    rebuilding said first branch prediction information in response to determining said second level cache contains said second branch prediction information, wherein said rebuilding comprises:
    <u>receiving said second branch prediction information;</u>
    <u>receiving a group of instructions corresponding to the fetch address;</u>
    <u>utilizing the second branch prediction information to identify one or more predicted taken branches within the group of instructions;</u>
    <u>generating third branch prediction information by decoding each of the identified one or more predicted taken branches to determine a type of each of the one or more predicted taken branches;</u>
    ~~generating third branch prediction information indicative of a type of branch instruction;~~ and
    combining said second branch prediction information with said third branch prediction information;

storing said combined second and third branch prediction information as said first

branch prediction information in a first entry of said first level cache,

wherein said first entry corresponds to said ~~first~~ fetch address.

2. (Original) The method of claim 1, further comprising:

determining if said first entry of said first level cache is available;

evicting contents of said first entry in response to detecting said first entry is not

available; and

storing a subset of said contents in said second level cache responsive to said

eviction.

3. (Currently Amended) The method of claim 1, wherein generating said third branch

prediction information further comprises determining a size of any immediate or

displacement data for each of the one or more predicted taken branches ~~said h~~

~~prediction corresponds to a first branch instruction, and wherein said branch~~

~~prediction further comprises information indicating a type of said branch instruction~~.

4. (Currently Amended) The method of claim 3, wherein generating said third branch

prediction information further comprises determining whether each of the one or

more predicted taken branches ends on an even addressed byte or an odd addressed

byte. ~~rebuilding said first branch prediction comprises decoding said branch~~

~~instruction.~~

5. (Original) The method of claim 4, wherein said branch instruction is fetched from

said second level cache.

6. (Original) The method of claim 1, wherein said subset comprises a dynamic bit.

7. (Original) The method of claim 6, wherein said subset further comprises a branch

marker bit.

8. (Original) The method of claim 7, wherein said branch prediction further comprises an end adjustment bit.

9. (Currently Amended) A branch prediction mechanism comprising:

a first level cache configured to store branch prediction information;

a second level cache configured to store a subset of said branch prediction information;

circuitry coupled to said first level cache and said second level cache, wherein said circuitry is configured to:

detect said first level cache does not contain a first branch prediction information corresponding to a ~~first~~ fetch address;

determine whether said second level cache contains a second branch prediction information corresponding to said ~~first~~ fetch address, said second branch prediction information comprising a subset of said first branch prediction information; and

rebuild said first branch prediction information in response to determining said second level cache contains said second branch prediction information, wherein in order to rebuild said first branch prediction information, said circuitry is configured to:

receive said second branch prediction information;

receive a group of instructions corresponding to the fetch address;

utilize the second branch prediction information to identify one or more predicted taken branches within the group of instructions;

generate third branch prediction information by decoding each of the identified one or more predicted taken branches to determine a type of each of the one or more predicted taken branches;

~~generate third branch prediction information indicative of a type of branch instruction;~~ and

combine said second branch prediction information with said third branch prediction information;

store said combined second and third branch prediction information as said first branch prediction information in a first entry of said first level cache, wherein said first entry corresponds to said ~~first~~ fetch address.

10. (Original) The mechanism of claim 9, wherein said circuitry is further configured to:

determine if said first entry of said first level cache is available;

evict contents of said first entry in response to detecting said first entry is not available; and

store a subset of said contents in said second level cache responsive to said eviction.

11. (Currently Amended) The mechanism of claim 9, wherein ~~said~~ generating said third branch prediction information further comprises determining a size of any immediate or displacement data for each of the one or more predicted taken branches ~~branch prediction corresponds to a first branch instruction, and wherein said branch prediction further comprises information indicating a type of said branch instruction.~~

12. (Currently Amended) The mechanism of claim 11, wherein generating said third branch prediction information further comprises determining whether each of the one or more predicted taken branches ends on an even addressed byte or an odd addressed byte. ~~rebuilding said first branch prediction comprises decoding said branch instruction.~~

13. (Original) The mechanism of claim 12, wherein said branch instruction is fetched from said second level cache.

14. (Original) The mechanism of claim 9, wherein said subset comprises a dynamic bit.

15. (Original) The mechanism of claim 14, wherein said subset further comprises a branch marker bit.

16. (Original) The mechanism of claim 15, wherein said branch prediction further comprises an end adjustment bit.

17. (Currently Amended) A computer system comprising:

an interconnect;

a memory coupled to said interconnect;

a second level cache configured to store branch prediction information;

a processor including a first level cache, wherein said processor is configured to:

detect said first level cache does not contain a first branch prediction information corresponding to a ~~first~~ fetch address;

determine whether said second level cache contains a second branch prediction information corresponding to said ~~first~~ fetch address, said second branch prediction information comprising a subset of said first branch prediction information;

rebuild said first branch prediction information in response to determining said second level cache contains said second branch prediction information, wherein in order to rebuild said first branch prediction information, said processor is configured to:

receive said second branch prediction information;

receive a group of instructions corresponding to the fetch address;

utilize the second branch prediction information to identify one or more predicted taken branches within the group of instructions;

generate third branch prediction information by decoding each of the identified one or more predicted taken branches to determine a type of each of the one or more predicted taken branches;

~~generate third branch prediction information indicative of a type of branch instruction;~~ and

combine said second branch prediction information with said third branch prediction information;

store said combined second and third branch prediction information as said first branch prediction in a first entry of said first level cache, wherein said first entry corresponds to said first address.

18. (Original) The system of claim 17, wherein said processor is further configured to determine if said first entry of said first level cache is available; evict contents of said first entry in response to detecting said first entry is not available; and store a subset of said contents in said second level cache responsive to said eviction.

19. (Currently Amended) The system of claim 17, wherein <u>generating said third branch prediction information further comprises determining a size of any immediate or displacement data for each of the one or more predicted taken branches</u> ~~said branch prediction corresponds to a first branch instruction, and wherein said branch prediction further comprises information indicating a type of said branch instruction~~.

20. (Currently Amended) The system of claim 19, wherein <u>generating said third branch prediction information further comprises determining whether each of the one or more predicted taken branches ends on an even addressed byte or an odd addressed byte.</u> ~~rebuilding said first branch prediction comprises decoding said branch instruction~~.

21. (Previously Presented) The method of claim 1, wherein said second level cache and said first level cache do not store duplicate information.

22. (Previously Presented) The mechanism of claim 9, wherein said second level cache and said first level cache do not store duplicate information.

23. (Previously Presented) The system of claim 17, wherein said second level cache and said first level cache do not store duplicate information.